Hamming Error Correcting Code

Michael Wimble
6026 Underwood Av SW
Cedar Rapids IA 52404

One of the most frustrating aspects of computers is that they make errors. Large computers have ample redundancy and error correcting hardware to make these errors virtually nonexistent, but owners of smaller computers must typically live with the problem. These errors are not all due to unreliable hardware; they are also caused by noisy environments, line crosstalk, power fluctuations, thermal variations and so on. To meet these problems several techniques have been developed. One of them is called Hamming codes.

The use of Hamming codes is analogous to the use of the common parity bit. A single parity bit merely provides detection of single bit errors, however. In contrast, the Hamming code described herein corrects single bit errors and detects double bit errors. An ideal use for Hamming codes is in cassette recording where single bit dropouts due to tape inconsistencies are common. Larger computers also use Hamming codes to detect and correct memory errors.

Alas, as with all real systems, you don't get something for nothing. To record eight bits of data without any error detection or correction scheme requires only eight bits of memory space. To use the common parity bit approach adds only one more bit of data: nine bits of memory space. But the Hamming code described here requires eight extra bits of memory space for every eight bits of data recorded. There are other Hamming codes available which use considerably less extra memory space per data space, but this one is particularly appropriate for microcomputers and for cassette recording, the most frequent source of errors in a typical hobbyist microcomputer system.

No attempt will be made to discuss the mathematics behind Hamming codes. The reader is referred to any good book on data transmission or error correcting codes for more information.

Building the Hamming Code

Building the Hamming code information is as simple as a table lookup. Every byte (eight bits) of data is recorded a nybble (four bits) at a time. Also, each group of four bits of data has four bits of Hamming data appended so that each byte of data to be recorded actually requires two bytes of storage.

Recording a byte of data is straightforward. Take the leftmost (most significant) nybble of data and record the corresponding byte of data shown in table 1. Next take the rightmost or least significant nybble of data and, again, record instead the corresponding byte from table 1.

For example, to record the data byte, hexadecimal 1F, perform the following:

- Extract the leftmost nybble (hexadecimal 01).
- Replace with corresponding byte from table 1 (hexadecimal E1).
- Record the byte.
- Extract rightmost nybble (hexadecimal 0F)
- Replace with corresponding byte from table 1 (hexadecimal FF).
- Record the byte.

Thus the single hexadecimal byte 1F is actually recorded as the two hexadecimal bytes E1FF. If it is not already obvious, each byte in table 1 has the actual data in its rightmost nybble and error correcting data in its leftmost nybble.

Reconstructing the Data

We are now ready to see how the Hamming code functions. As each byte of stored data is retrieved, with or without errors, four parity bits are constructed which give information as to the correctness of the retrieved byte. Using these parity bits, any required corrections are made to the re-

The timer processor scans the file of delay timer records (table 7) to process those records which are active. The function of the delay timer record is to provide a delay of a preset number of seconds before processing an event record. To demonstrate how this function works, let's assume that one wishes to activate an audible alarm 50 seconds after a particular sensor has been tripped. You would structure the event and response records so that the tripping of the sensor would activate the delay timer record you were associating with this event. The activation of the delay timer causes the active flag to be reset and the timer activation value (bytes 2 and 3) to be transferred to the delay time. In our example this will cause the value 50 to be loaded into bytes 0 and 1 of our delay timer value.

As each real time clock interrupt causes the timer processor to scan the file of delay time records, it will find that our record is active. Once the processor determines that a record is active it will decrement the current delay time remaining and then check to see if the time remaining is zero. If there is still delay time remaining (time greater than zero) no further action is taken.

However, if there is no time remaining (time equals zero) the active flag is set, the current registers saved, the address of the event record is extracted from the delay time record, and control transferred to the event processor. The sequence of operations performed from this point will directly result in the audible alarm being turned on.

The processing of the records in the time of day file (see table 6 for record format) is performed upon the completion of all delay time records. Time of day records are processed in a manner very similar to that described for delay timer records. When an active record is encountered during the scan of the time of day file, the current time of day being maintained by the system will be compared to the time of day specified in bytes 0 and 1 of the record. Should these times be identical, the records flag is set to 1 and the processing of the event record specified in bytes 2 and 3 is initiated as described above. After servicing all records in the time of day file, control is returned to the interrupted modules.■

Next Month: *Part 3 of the Computerized Security System illustrates the design of some of the sensors, gives listings of a few of the control modules, and describes the design of the remote sensor display panel.*

Circle 72 on inquiry card.

NOBODY SELLS THE BEST FOR LESS!

COMPUTER LAB OF NEW JERSEY

	LIST PRICE	SPECIAL PRICE
Solid State Music 8080 CPU Board Kit	$149.95	$127.45
Mountain Hardware Introl System	$329.00	$279.00
IMC Keyboard	$169.00	$143.65
All Versions of Electric Pencil	15% off	
Problem Solver – All Products	15% off	
Vector Graphics All Products	15% off	
Imsai – All Products	15% off	

SUBJECT TO AVAILABLE QUANTITIES. SHIPPING AND INSURANCE EXTRA. SHIPPING FREE ON PREPAID ORDERS.

COMPUTER LAB OF NEW JERSEY
141 ROUTE 46
BUDD LAKE, NEW JERSEY 07828
(201) 691-1984

SURPLUS PRINTER FOR TRS80*
OKIDATA CP110
LINE PRINTER
WITH INTERFACE
BOARD.

*T.M. - RADIO SHACK

NO SOFTWARE OR HARDWARE CHANGES REQUIRED. JUST PLUG IN AND RUN!

- 5x7 Impact Dot Matrix
- 80 Char/Line
- 64 Char ASCII (Upper Case)
- 110 Char/Sec.
- 66 Lines/Min.
- Accepts 8-1/2" TTY Roll paper

PRINTER $650.00

INTERFACE: BUILT $100.00
KIT $ 60.00
INFOR & SCHEMATIC $ 5.00

Shipped Freight collect. Send check, M.O.; VISA* master charge

INCLUDES - Power Supply, Built in Selftest, Parallel Interface, Line Buffer and Cables. Housed in a three piece plastic cabinet with all control electronics. Retail for over $1,100. PRINTER BRAND NEW NEVER USED IN FACTORY SEALED CARTON. Operating Manual Included. *Supplies Limited*

Guaranteed to be in good working order at time of delivery.

Write for Interface Info on Heath, Apple, Imsai, Sol, Northstar

INTERNATIONAL ELECTRONICS EQUIPMENT CORP.
P.O. Box 522542, Miami, Florida 33152

Circle 188 on inquiry card.

Data Nybble	Hamming Byte
0	00
1	E1
2	72
3	93
4	B4
5	55
6	C6
7	27
8	D8
9	39
A	AA
B	4B
C	6C
D	8D
E	1E
F	FF

Table 1: Table for transforming data nybbles into Hamming code bytes for storage.

trieved byte after which the original nybble of data is extracted.

Parity bit 4 (P4), the most significant parity bit, is simply the parity of the entire bit retrieved byte. If the 8 bit data word nas an odd number of 1s, then P4 is set to 1. Similarly, if the 8 bit data word has an even number of 1s, P4 is set to 0. To form parity bit 3 (P3), the 8 bit data word is first ANDed with hexadecimal 27 and the 8 bit parity of the result determines P3 just as P4 was calculated. Likewise parity bit 2 (P2) is formed by ANDing the retrieved data word with hexadecimal 4B; parity bit 1 (P1), the least significant parity bit, is formed from ANDing the retrieved data word with hexadecimal 8D. Table 2 shows a sample calculation of the parity bits when hexadecimal is retrieved.

Error detection and correction is performed by detecting one of three cases:

- If all four parity bits are 0, the retrieved data word is correct. The original nybble of data is correctly formed in the rightmost nybble of the retrieved data word.
- If P4 is 0 but one or more of the parity bits P1, P2, or P3 is not 0, then a double bit error has occurred. The program should probably inform the operator and then halt.
- If P4 is not 0, a single bit error has occurred, which can be corrected.

To correct single bit errors, a byte chosen from table 3 is exclusive-ORed with the retrieved data byte. Parity bits P3, P2, and P1 (P3 is most significant and P1 is least significant) determine which byte to select from table 3. Table 4 demonstrates the correction and detection process for several different cases.

Data Word	AND	Result	Parity Bit
01100111	—	01100111	P4 = 1
01100111	00100111	00100111	P3 = 0
01100111	01001011	01000011	P2 = 1
01101111	10001101	00000101	P1 = 0

Table 2: A sample calculation of parity bits for the binary data word 01100111 (hexadecimal 67). Since P4 is not 0, a single bit error has occurred, which can be detected (see text).

P3	P2	P1	Error Correction Byte
0	0	0	10
0	0	1	80
0	1	0	40
0	1	1	08
1	0	0	20
1	0	1	04
1	1	0	02
1	1	1	01

Table 3: Single bit error correction table.

Case 1: No errors.

Data	AND	Parity	Comments
E1	—	P4 = 0	All parity bits 0 implies no error.
E1	27	P3 = 0	
E1	4B	P2 = 0	
E1	8D	P1 = 0	

Case 2: Single bit error.

Data	AND	Parity	Comments
E3	—	P4 = 1	P4 not 0 implies single bit correctable error.
E3	27	P3 = 1	P3P2P1 = 110
E3	4B	P2 = 1	Corrector from table 3 = 02
E3	8D	P1 = 0	Data E3; Exclusive OR 02; Correct data E1

Case 3: Single bit error.

Data	AND	Parity	Comments
C1	—	P4 = 1	P4 not 0 implies single bit correctable error.
C1	27	P3 = 1	P3P2P1 = 100
C1	4B	P2 = 0	Corrector from table 3 = 20
C1	8D	P1 = 0	Data C1; Exclusive OR 20; Correct data E1

Case 4: Double bit error.

Data	AND	Parity	Comments
E2	—	P4 = 0	P4 equal to 0 and P3, P2, or P1 not 0 implies a double bit error. Note: data in P3, P2, and P1 for a double bit error does not imply any information about which bits are in error.
E2	27	P3 = 0	
E2	4B	P2 = 0	
E2	8D	P1 = 1	

Table 4: Examples of uses of the Hamming code. For each case the transmitted information is hexadecimal E1. When a 1 bit error is detected, the parity bits P3, P2, and P1 are used to look up a correcting factor from table 3.

Circle 396 on inquiry card.

get your hands on...

Hands on microprocessor short course with **FREE** take home microcomputer included in the $499 tuition.

March 5, 6, 7, 8, 9	Atlanta, GA
March 19, 20, 21, 22, 23	Lafayette, IN
April 2, 3, 4, 5, 6	Los Angeles, CA
May 7, 8, 9, 10, 11	Boston, MA
June 11, 12, 13, 14, 15	New York, NY

Learn microprocessors first hand from the original hands on people.

For more information call Jerilyn Williams, (317) 742-6802 or write Wintek Corp., 902 North 9th Street, Lafayette, Indiana 47904.

- **6800 Hardware/Software**
- **Custom Hardware/Software**
- **In-house short courses**

Conclusion

The algorithm described here has been programmed on several different microcomputers. Not many bytes of program are actually needed and the benefits are great. One need only read a large program from tape into main memory several times to realize the utility of the approach. If your current read in software informs you of any errors encountered, you probably must still find the errors and correct them, assuming you even know what the correct data should be.

Hamming codes can be extended to correct double bit errors, or to perform any practical n bit detection and m bit correction, but the extra memory costs can climb fast. Also it is very easy to build a hardware Hamming generator and detector using only a few discrete integrated circuits.

The Hamming code described here is both practical and valuable. Manufacturers should seriously consider incorporating this technique in hardware. The cassette enthusiast should incorporate this technique in any standardization or interchange effort. Even the everyday experimenter will find that the hour spent programming the algorithm will save many hours of frustration in the future.■

INFO 2000 BUSINESS SYSTEMS with TEXT 2000 Word Processing and CPA 2000 Business Accounting

Professional Equipment

- Z80A cpu, 32K RAM and PerSci 8" dual drives.
- High speed line printer*, 150 characters per second.
- Word processing printer* with metal print wheel.
- Video terminal with capacitive keyboard, 24 x 80 display.

*Choice of one.

TEXT 2000™ Word Processing

- True proportional spacing
- Full-screen text editing with simple commands.
- Bidirectional, bold face, underlining, tab, right and left justification, centering, auto. pagination, name & address list.

CPA 2000™ Business Accounting

- Accounts Receivable
- Accounts Payable
- General Ledger –(multi-co.)
- User defined formats for bal. sheet, income statement, changes in financial position.
- Integrated A/R, A/P, G/L.

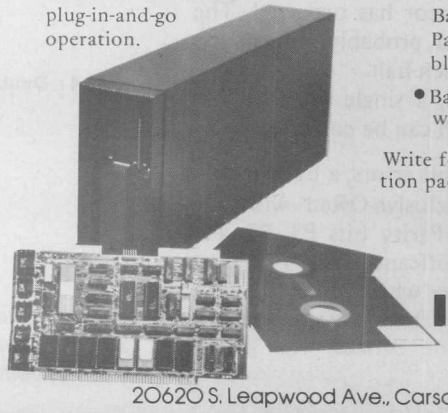
Complete program development software – Basic, Fortran, Cobol, Pascal, macro-assembler, debugger–available. INFO 2000 Business System, complete with TEXT 2000 and CPA 2000 software, can be purchased for less than $13,000. 90-day warranty with system. Extended warranty available. Write for complete information packet. Dealer inquiries welcome.

INFO 2000 CORPORATION
20620 South Leapwood Avenue Carson, California 90746 (213) 532-1702

INFO 2000 DISK SYSTEMS For: Z80, 8080, 8085 S-100, Digital Group, Heath H8

COMPARE THESE UNIQUE FEATURES:

- PerSci Dual Disk Drives
- Full size 8" floppy diskettes.
- $hundreds less than comparable systems.
- Intelligent controller boards for each type of bus increases computer capabilities.
- Additional I/O ports and drivers.
- CP/M* Disk Operating System.
- Voice-coil positioning 8 times faster than others.
- IBM 3740 compatible.
- Takes less than 1/2 the space of others.
- Includes cabinet, power supply, fan and cables.
- Factory assembled and tested for immediate plug-in-and-go operation.
- Extensive software library available: Basic, Fortran, Cobol, Pascal, macro-assembler, debugger.
- Backed by 90-day warranty.



Write for complete information packet. Dealer inquiries welcome.

*Trademark of Digital Research

INFO 2000 CORPORATION
(213) 532-1702
20620 S. Leapwood Ave., Carson, California 90746

Circle 178 on inquiry card.